2.3 PLATFORM SIGNATURE

In Suppressor, an element is the part of a player which can be detected and engaged with lethal weapons. Therefore all signature attributes in Suppressor belong to the elements of players. There are four distinct types of sensors in Suppressor and each sensor type is able to detect a distinct type of signature data summarized in Table 2.3-1.

TABLE 2.3-1. Suppressor Sensor Types.

Sensor Type	Signature Type	Signature Data Item
Radar	Radar Cross Section	RCS-TABLE
Infrared	Infrared Signature	IR-INTENSITY
Optical	Optical Cross Section	OPT-CS
Warning Receiver	Detects radar or communications transmitter system emissions.	

In Suppressor, the signature of a target can change dynamically during a simulation by using the configuration dimension in a signature table and a change in configuration action in the tactics of the target player. This allows different target configurations such as bays-open, bays-closed, external-stores, or no-external-stores to be associated with different sets of signature data; the different signatures associated with each configuration are utilized at appropriate times during the simulation.

2.3.1 Functional Element Design Requirements

Design requirements for the signature functional element are:

a. Provide a capability for the user to define radar cross section (RCS) for each element of a player. The RCS will be given in units of area and may vary in a step-wise manner for any number of frequency, azimuth, elevation, and polarization intervals of any (possibly irregular) sizes defined by the user.

b. Provide the capability for the user to define infra-red (IR) signatures for each element of a player. The IR signatures will be given in units of radiant intensity (watts/steradian) and may vary in a step-wise manner for any number of frequency, azimuth, and elevation intervals of any (possibly irregular) sizes defined by the user.

c. Provide the capability for the user to define optical signatures for each element of a player. The optical signatures will be given in units of area and may vary in a step-wise manner for any number of azimuth and elevation intervals of any (possibly irregular) sizes defined by the user.

d. The IR signature must include the effects of solar irradiance and background contrast.

e. The RCS and IR signature data must include a capability to change dynamically to represent different platform configurations such as bay doors open and closed, with and without external weapons or fuel stores, and with and without afterburner.

2.3.2 Functional Element Design Approach

Design Element 3-1: Static Signature

Each signature table may include some optional dimensions in addition to the azimuth and elevation dimension. Table 2.3-2 summarizes the dimension types allowed in each signature table.

TABLE 2.3-2. Allowable Dimensions for Signature Data.

Table	Allowable Dimensions	Required / Optional
RCS-TABLE	CONFIGURATION-TYPES FREQ POL AZ EL	Optional Optional Optional Required Required
IR-INTENSITY	CONFIGURATION-TYPES IR-BAND AZ EL	Optional Optional Required Required
OPT-CS	CONFIGURATION-TYPES AZ EL	Optional Required Required

CONFIGURATION-TYPES is the dimension name used to define different configurations of signature, such as bays-open and bays-closed. This dimension facilitates the dynamic changes in signature during a simulation.

FREQ is the dimension name for the operating frequency of a radar sensor. POL is the dimension name for the polarization of the radar sensor.

IR-BAND is the dimension which allows differences in infrared signature for sensors operating in different infrared bands.

AZ and EL are the dimension keywords for the azimuth and elevation dimensions, respectively. The zero azimuth and elevation angle of each element is aligned with the heading vector of its parent location. If the player is moving, then the heading vector is aligned with the velocity vector, with an addition for angle of attack if angle of attack is included for the mover. If the player is stationary, then the heading vector is aligned with a heading specified in the SDB, the most recent cuing order, or East as a default.

In addition to the signature dimensions, Suppressor allows several orderings of the dimensions to be specified. For example, the AZ may be followed by the EL dimension, or the EL dimension may be followed by the AZ dimension. Further, the number of entries or resolution for each dimension can be defined. This means that signature data may be defined with very fine (e.g. 1-degree resolution in azimuth and elevation) through very coarse resolution.

Design Element 3-2: Dynamic Signature

Dynamic target signature changes in Suppressor are an extension of the static signature data and algorithms. Dynamic signature requires 1) the addition of a CONFIGURATION-TYPES dimension in the appropriate signature data items, 2) the definition of an INITIAL-CONFIGURATION for the target element, and 3) the addition of appropriate "NOW-USE CONFIGURATION" actions in the MOVE-PLANS tactics or "WITH CONFIGURATION" phrases in the RESOURCE-ALLOCATION tactics.

The code to implement dynamic signatures stores the initial configuration for each element, and changes it whenever one of the configuration change actions is executed as part of a player's tactics. The signature look-up code described below in the 2.3.3 Static Signature Module Design (module SIAREA) determines the current configuration for a target element whenever a signature table includes a CONFIGURATION-TYPES dimension.

2.3.3 Functional Element Software Design

Static Signature Module Design

Signature data in Suppressor are defined in the TDB using the RCS-TABLE, IR-INTENSITY (or IR-RAD-TABLE, a synonym) and OPT-CS tables. Each of these tables includes some optional dimensions, but the last two dimensions must be azimuth and elevation around the target element. Since the signature data are defined entirely in input tables, the code for determining signatures is primarily a table look-up algorithm. There are two software modules that perform the table look-ups are for radar cross-section (RCS) data and one for Infra-red (IR) and optical signatures. The two subroutines and the relevant TDB signature data items are listed in Table 2.3-3.

TABLE 2.3-3. Signature-related Software Modules.

Modules	Data Block (number)	Data Items
SIAREA SILUET	Radar Signature (150) Infrared Signature (150) Optical Signature (150)	RCS-TABLE IR-INTENSITY OPT-CS

The signature look-up software is implemented primarily in two modules, SILUET and SIAREA. SILUET is used to perform most of the initial computations; it then invokes SIAREA to perform the actual signature table look-up.

Subroutine SILUET performs two functions, determining the cross section of a target and pointing the sensor depending on its slew limits and mode. (The software design for the sensor pointing is omitted in this functional element description.) This is the initial part of the signature look-up performed in SILUET:

```
ABSTRACT     compute cross sectional area of target and point sensor
   *begin logic to compute cross section of target:
      *look up vector pointers;
      *look up target, sensor cluster pointers, type of sensor;
      *get pointer to location for target, receiver;
      *look up target and receiver positions and directional vectors;
      *invoke logic to normalize range vector;
```

```
      *initialize signature vector using only receiver position;
      *when there is a sensor transmitter:
         *calculate range vector and normalize;
         *calculate bistatic vector;
      *end of test for transmitter.
      *invoke logic to adjust UXT and UZT for angle of attack;
      *calculate relative angles with respect to bistatic vector;
      *loop, until all target groups are checked for detection:
         *when detection possibility is defined:
            *when not a warning receiver:
               *invoke logic to get cross section per sensor type;
               *when infrared sensor:
                  *invoke logic to get optical cross section;
                  *look up reflectivity value for element;
                  *when constant part of signature not computed:
                     *look up solar irradiance based on target alt;
                     *compute constant part of signature;
                     *look up background radiance at target;
                  *end of test for constant part of signature.
                  *calculate ir pixel intensity difference;
               *end of test for infrared sensor.
               *when optical sensor:
                  *combine effects of contrast and area;
               *end of test for optical sensor.
               *determine element possessing largest signature;
            *end of test for warning receiver.
         *end of test for detection possibility defined.
      *end of loop for target groups.
       .
       .
       .
*end of logic for SILUET.
```

This is the remainder of the signature look-up performed in SIAREA:

```
ABSTRACT     look up radar, optical, or infrared cross section
   *begin logic to look up cross section component of target:
      *look up pointer to cross section table;
      *when configuration type dependent:
         *search for configuration type or default;
      *end of test for configuration type dependent.
      *loop for frequency/polarization dependency in either order:
         *when frequency dependent cross section used (radars):
            *determine appropriate frequency interval;
         *end of test for frequency dependency.
         *when polarization dependent cross section used (radars):
            *determine appropriate polarization interval;
         *end of test for polarization dependency.
      *end of loop for frequency/polarization dependency.
      *when band dependent cross section used (ir):
         *determine appropriate frequency band;
      *end of test for frequency band dependency.
      *when az/el cross section table:
         *look up cross section and get effective area;
      *otherwise, el/az cross section table:
         *look up cross section and get effective area;
      *end of test for cross section dimension order.
   *end of logic for SIAREA.
```

Dynamic Signature Module Design

Two modules, CONFIG and RUNEXE, are used to implement dynamic configuration changes. Subroutine CONFIG is invoked after every RESOURCE-ALLOCATION

decision to implement any WITH CONFIGURATION phrase associated with the current tactical decision. This is the software design for Subroutine CONFIG:

```
ABSTRACT     process any with-configuration phrases
   *begin logic to process "with-configuration" phrases:
      *look up pointers;
      *loop, while more "with" phrases:
         *when "with-configuration":
            *when element specifically mentioned in "with-" phrase:
               *loop, for each location:
                  *search for specified element;
               *end of loop for each location.
            *otherwise:
               *store element containing resource system;
            *end of test for element specifically mentioned.
            *when element found:
               *when new configuration different from old:
                  *store user id for element;
                  *find ptr to character description (246) for element;
                  *find ptr to character description (246) for configs.;
                  *store new configuration;
               *end of test for new configuration different from old.
            *end of test for element found.
         *end of test for "with-configuration".
      *end of loop for more "with" phrases.
   *end of logic for CONFIG.
```

Subroutine RUNEXE is invoked to implement any MOVE-PLANS actions associated with the current reactive movement decision. The software design for the subset of Subroutine RUNEXE which implements the configuration change caused by a NOW-USE CONFIGURATION action follows:

```
ABSTRACT     execute any operations related to movement
   *begin logic to execute movement-related operations:
       .
       .
       .
      *when configuration change:
         *when specific element mentioned:
            *loop, for each location:
               *search for specified element;
            *end of loop for each location.
         *otherwise:
            *store element containing mover system;
         *end of test for specific element mentioned.
         *when element found:
            *when new configuration different from old:
               *store user id for element;
               *find ptr to character description (246) for element;
               *find ptr to character description (246) for configs.;
               *store new configuration;
            *end of test for new configuration different from old.
         *end of test for element found.
      *end of test for configuration change.
   *end of logic for RUNEXE.
```

2.3.4 Assumptions and Limitations

No interpolation of the signature data tables are performed. It is assumed that the user has defined the table for the requisite resolution of the scenario being modeled.

2.3.5 Known Problems or Anomalies

None.